FIRE GROWTH SIMULATION MODEL

BY

Collin D. Bevins
Patricia Andrews

FINAL REPORT FOR AGMT #INT-89417-RJVA
with SYSTEMS FOR ENVIROMENTAL MANAGEMENT

FS Contact: Patricia L. Andrews

Sem Contact: Collin D. Bevins
(3 parts)

FIRE GROWTH SIMULATION MODEL

FINAL REPORT

INT-89417-RJVA

Collin D. Bevins
Systems for Environmental Management
Missoula, MT 59807

in cooperation with

Patricia Andrews
Intermountain Research Station
USDA Forest Service

1. Study Objectives

The objectives of the Fire Growth Simulation Model study are:

> "... to develop a prototype fire growth simulation model linked to a GIS. The simulation may include place holders for some of the needed models. But it should be a complete working system with fuel and terrain that vary over space, and wind and fuel moisture that vary in both time and space. The simulation should include both surface and crown fire models, and the transition from one to the other."
> - (Research Joint Venture Agreement No. INT-89417-RJVA, May 1989)

This report briefly describes the characteristics, capabilities, and limitations of Gfire Version 1.0, the result of the 6 month study. Two companion reports, User's Guide to Gfire Version 1.0: A Fire Growth Simulator for GRASS 3.0, and Technical Documentation to Gfire Version 1.0: A Fire Growth Simulator for GRASS 3.0 specifically address the use and structure of Gfire.

Gfire 1.0 is a first generation product demonstrating the feasibility of fire growth modelling within a GIS environment. Gfire 1.0 is a work in progress with many limitations, yet it reveals great potential for refinement, enhancement, and expansion.

Gfire 1.0 is a research and development tool only. It is not, in its current state, suitable for wildland fire planning and evaluation task!

2. Study Methods

2.1 GIS Selection

The major objective of the study was to design and implement a prototype fire growth simulator closely integrated with a geographic information system. GRASS 3.0 was selected as the GIS because:

- o it is a complete and robust GIS,
- o it is in the public domain,
- o it is actively maintained by the U.S. Army Corps of Engineers, CERL,
- o it is widely used and supported by quarterly publications and annual user meetings,
- o it is ported (and portable) to a number of hardware platforms running the ubiquitous UNIX operating system,
- o it is used by the National Park Service, who provided the Yellowstone National Park GIS database for the study,
- o and most importantly, it has an open architecture with published interface libraries (written in the C programming language) supporting third party software development and integration.

2.2 Development Environment

Gfire was written in the ANSI standard C programming language. Only those ANSI standard functions available on SUN UNIX were used.

Gfire code was first developed, compiled, linked, executed, and debugged using Turbo C Professional (tm) on a PC class computer running the MS-DOS operating system. The code was then transmitted onto the SUN 4 which hosted the GRASS 3.0 GIS. The code was then recompiled, linked, executed, and tested under UNIX and GRASS.

The benefits of this approach were:

- o faster code development time using wysiwyg text editors, integrated compilers, and integrated debuggers on the PC,
- o more stable code resulting from successful compilation, linking, and testing in two hardware and software environments, and
- o portable source code (Gfire source code is identical for the two environments).

3. Gfire 1.0 Capabilitiess

o Gfire simulates fire growth in a fuel array of arbitrary complexity. Fuel type information is accessed directly from GRASS 3.0 cell maps. Fuel type cell maps may be created, modified, analyzed, and displayed just as any other cell map via GRASS 3.0 tools.

o Fuel cell maps may reference up to 255 fuel models (plus a "no fuel" model). Model parameters are stored in an ASCII text file created and maintained by the user. Standard and/or custom fuel models may be used, and fuel models may contain any number of dead and live fuel particles.

o Gfire simulates fire growth in terrain of arbitrary complexity. Elevation information is obtained directly from GRASS 3.0 cell maps and used to determine slope between fuel cells.

o Gfire obtains the initial (time 0) fire pattern directly from a GRASS 3.0 cell map. It is possible to start fires from a point, line, or area source, or to mix ignition patterns of arbitrary complexity.

o Gfire responds to changes in weather conditions over time. Weather observations are obtained from an ASCII text file created and maintained by the user. Observations may occur at any frequency, and include the day, hour, wind speed and direction, and fuel moisture contents. Gfire 1.0 accepts only a single weather stream which is applied to the entire simulation window.

o Gfire simulates fire growth through ground fuels using the "cell escape time" concept. Once a cell is ignited, it must wait a certain amount of time before it can escape to one of its unignited neighbors. This escape time is dependent upon the cell's fuel type and moisture contents, and upon the effective wind and slope in each neighbor's direction. Gfire appropriately adjusts escape times when weather conditions change.

o Gfire simulates fire growth until (1) all ignitable cells have ignited, (2) the fire reaches a user specified size (expressed as cells or acres), or (3) the fire burns for a user specified time (expressed as days, hours, minutes, or seconds).

o Gfire produces a GRASS 3.0 cell map containing ignition times. Unignited cells have an ignition time class of 0. The user controls ignition time class size.

4.0 Proposed Enhancements

Gfire 1.0 is a highly modular program constructed to facilitate modifications, enhancements, and additions. The following enhancements would evolve Gfire 2.0 closer to a real fire management application tool.

- o Accept weather observations from a network of reporting stations. Station characteristics would include location and elevation.
- o Modify weather observations to include day, time, wind speed, wind direction, temperature, and relative humidity (or equivalent).
- o Develop and/or incorporate algorithms to determine fuel moistures within cells based upon (1) weather conditions interpolated spatially and temporally from the weather station network, (2) cell elevation, slope, and aspect, and (3) cell fuel and canopy type.
- o Develop and/or incorporate algorithms to determine wind speed and direction within cells based upon current weather conditions and terrain influences.
- o Accept additional layers describing canopy closure, height, density, or whatever is required to model (1) sheltering effects on fuel moisture, (2) wind speed at midflame height, (3) ground-to-crown transition probabilities, and (4) crown fire spread.
- o Develop and/or incorporate algorithms to determine ground-to-crown transition probabilities within map cells.
- o Develop and/or incorporate algorithms to determine crown fire spread within and between map cells.
- o Create additional output map products for derived values such as
  - interpolated weather conditions at given time intervals,
  - derived fuel moistures at given time intervals,
  - derived wind speed and direction at given time intervals,
  - fire intensity at given time intervals,
  - ground-to-crown fire transition probabilities at given time intervals,
  - flame length at given time intervals,
  - fire scorch height at given time intervals, and
  - estimated tree mortality.
- o Display fire growth on the console during simulation.
- o Investigate alternative fire contagion algorithms to reduce fire shape distortion and to improve system performance.

Gfire 3.0 could add the ability to interact with the console during fire simulation. This would permit users to build fireline and backfire in response to changing weather and fire conditions.

5. Conclusions

Gfire 1.0 demonstrates the feasibility of incorporating fire behavior inputs and algorithms into a general purpose geographic information system. As a first generation prototype, it is not ready for real fire management applications. Yet version 1.0 reveals its potential for integrating complex terrain, weather, biological, and fire behavior relationships into a manageable system accessible to fire scientists and trained fire management specialist.

6. Acknowledgements

This research was supported in part by funds provided by the Intermountain Research Station, Forest Service, U.S. Department of Agriculture (Agreement No. INT-89417-RJVA), and by the Bureau of Land Management, U.S. Department of Interior (Agreement No. AA851-IAO-77).

Additional funding and support was provided by Systems for Environmental Management, a Missoula, Montana nonprofit research and educational organization.

Gfire Version 1.0

A Fire Growth Simulator for GRASS 3.0

Users' Guide

November 1989

Collin D. Bevins
Systems for Environmental Management
P.O. Box 8868
Missoula, MT 59806

0. Table of Contents

1.0 Introduction

Gfire 1.0 is a first attempt to integrate fire behavior simulation with a general purpose geographic information system (GIS). The development objectives for Version 1.0 were to incorporate GIS based fire behavior inputs (fuel type, terrain, and ignition source maps) with standard and custom fuel models, temporal weather observations, and fire behavior algorithms to produce maps of fire growth over time.

Gfire 1.0 uses the GRASS 3.0 GIS developed by the U.S. Army Corps of Engineers. GRASS 3.0 is a public domain software system written in C for the UNIX operating system. Its open architecture and published interface libraries promotes the development of GIS extensions such as Gfire 1.0.

The development philosophy of Gfire 1.0 was to use the capabilities of GRASS 3.0 to create, modify, display, and summarize fuel type, terrain, and ignition source input maps and fire behavior output maps. The Gfire program reads and writes the maps via the GRASS 3.0 interface libraries. Gfire also accesses a file of fuel model parameters and a file of weather observations.

Gfire 1.0 is a first generation product demonstrating the feasibility of fire growth modeling within a GIS environment. While version 1.0 has obvious limitations, it reveals the potential for expansion and refinement.

Gfire 1.0 is a research and developmental tool only. It is not to be used for fire planing or real time fire evaluation tasks!

2.0 Gfire 1.0 Capabilities and Limitations

Gfire 1.0 has the following operational characteristics:

- o Fire growth projections are influenced by an underlying fuel mosaic of arbitrary complexity. The fuel map may be of any size or resolution (within system memory limitations). Fuel type maps contain integers in the range 0-255 corresponding to a fuel model described in a fuel model parameter file. The fuel map is specified in the Gfire input file by the "fuels=" command.
- o Fuel models referenced by the fuel map are defined in a fuel model parameter text file. The file may contain standard and/or custom fuel models, and a model may have any number of live or dead fuel particles. The fuel model parameter file is specified in the Gfire input file by the "models=" command.
- o Fire growth projections are influence by the underlying terrain. The terrain map may be of any size or resolution (within system memory limitations). Terrain maps contain integers representing elevation in 100-foot intervals. The terrain map is specified in the Gfire input file by the "elevation=" command.
- o Fire initiation depends upon an ignition source map. Map cells containing non-zero ignition times are used to initiate fire growth. A cell with value 'x' ignites at time interval 'x' (if it isn't already ignited from another source during simulation). The fire ignition source map is specified in the Gfire input file by the "ignition=" command.
- o Fire growth projections are influenced by a single weather stream which varies over time. Weather inputs are read from a text file containing the day, hour, wind speed and direction, and fuel moisture contents for 5 fuel classes. Weather observations may occur at any frequency. The weather observation file is specified in the Gfire input file by the "weather=" command.
- o Fire growth is based upon the cell "escape time" concept. Once a cell is ignited, the time required for it to escape to any of its eight neighbors depends upon (1) the cell's fuel type, (2) slope from the cell to its neighbor, (3) effective wind speed in the neighbor's vector, and (4) current fuel moisture.
- o Fire growth continues until (1) all ignitable cells have ignited, (2) the fire reaches a user specified size (expressed as cells or acres), or (3) the fire burns for a user specified time (expressed as days, hours, minutes, or seconds). The fire growth termination condition is specified in the Gfire input file by the "quit=" command.
- o Gfire produces a GRASS 3.0 layer of fire ignition times whose cell values indicate the time period during which the cell ignited. The Gfire input file "period=" command specifies the ignition time class size, and the "spread=" command names newly created growth map.

3.0 Preparation

GRASS 3.0 must be installed before using Gfire. You must have a GRASS Location available for your use.

Gfire requires inputs from 6 sources:

- o You must create a Gfire command file. It is a text file directing Gfire where to find its fuel, elevation, ignition source, model parameter, and weather information.
- o You must have a fuel model parameter file available on the system.
- o You must have a fuel layer available in your GRASS location and mapset path. This can often be created within GRASS 3.0 by reclassifying a vegetative cover layer. The fuel map should cover or exceed the area of the current window.
- o You must have an elevation layer available in your GRASS location and mapset path. Elevation classes must reference 100 foot intervals. The elevation map should cover or exceed the area of the current window.
- o You must have an ignition source layer available in your GRASS location and mapset path. This is easily created by first creating a site list and converting it to a cell map. The ignition source map may (preferably) be smaller than the current window.
- o You must have a weather observation file available on the system. It is a text file with one line per observation. At least 1 observation is required.

4.0 Starting Gfire

Before starting Gfire, you must start GRASS 3.0 using the "GRASS" command. After selecting the GRASS location and mapset, you may start Gfire.

Gfire is started by entering any one of the following command forms:

```
% Gfire commands=<command_file>

% Gfire input=<command_file>

% Gfire c=<command_file>

% Gfire i=<command_file>
```

where % is the UNIX system prompt and <command_file> is the name of the Gfire input command file.

5.0 Gfire Command File

The Gfire command file is a standard ASCII text file containing 6 to 8 records. Each record contains 2 sections: (1) a lower case command immediately followed by an "=" sign (2) immediately followed by a value and/or name. It is important that no spaces appear between the command and the "=", or between the "=" and the first character of the value section.

Below is a sample Gfire command file:

```
fuels=se_corner_fuels
elevation=elevation
models=/usr/grass3/data/yell/cbevins/fmodels.txt
weather=/usr/grass3/data/yell/cbevins/weather.txt
ignition=se_corner_ignition
spread=spread
period=1 minute
quit=1 days
```

5.1 fuels=

The fuels= command specifies the layer to be used as the fuel map. The layer must be present in the current GRASS location and mapset path and should span the current window.

The fuels layer must contain integers in the range 0-255 and each integer must reference a fuel model in the fuel model parameter file.

At least one fuels= command is required. If more than one fuels= command is present, Gfire uses the last one.

5.2 elevation=

The elevation= command specifies the layer to be used as the terrain map. The layer must be present in the current GRASS location and mapset path and should span the current window.

The elevation layer must contain integers referencing 100 foot elevation classes, otherwise slope calculations will be in error.

At least one elevation= command is required. If more than one elevation= command is present, Gfire uses the last one.

5.3 models=

The models= command specifies the fuel model parameter file to associate with the fuel layer. The file's entire path name must be entered unless it resides in the current working directory.

The fuel model parameter file is an ASCII text file. It must have a fuel model for every fuel type in the fuel map (including fuel map cells with value 0). Section 7 describes the fuel model file format.

At least one models= command is required. If more than one models= command is present, Gfire uses the last one.

5.4 ignition=

The ignition= command specifies the layer to be used as the fire ignition source map. The layer must be present in the current GRASS location and mapset path.

The ignition layer must contain integers indicating cell ignition times. For Gfire 1.0, cells that initiate the fire should have a value of 1, while all other cells have a value of zero.

At least one ignition= command is required. If more than one ignition= command is present, Gfire uses the last one.

5.5 weather=

The weather= command specifies the weather observation file. The file's entire path name must be entered unless it resides in the current working directory.

The weather observation file is an ASCII text file containing one record for every weather observation. A weather observation record contains the day, hour, wind speed, wind direction, fuel moisture contents for 0-.25", .25-1", and 1-3" dead fuels and fuel moisture contents for herbaceous and woody live fuels. Section 8 describes the weather observation file format.

At least one weather= command is required. If more than one weather= command is present, Gfire uses the last one.

5.6 spread=

The spread= command names the fire growth map to be created. The map layer will be stored in the current GRASS location and mapset.

The growth layer contains ignition time class values. Cells with a value of zero were unignited. Time class size is determined by the period= command. A sample fire growth map is shown in Section 9.

At least one spread= command is required. If more than one spread= command is present, Gfire uses the last one.

5.7 period=

The period= command specifies the output fire growth map's time class size. The following options are recognized:

```
period=<n> seconds
period=<n> minutes
period=<n> hours
period=<n> days
```

where <n> is a nonzero positive integer. A fire growth map cell with a zero value is unignited. Cell value 1 indicates the cell ignited in the first time period. For example, if

```
period=15 minutes
```

is specified, then a cell with value 4 was ignited during the 45-60 minute interval after the start of simulation.

If no period= command is present in the command file, the default period of 1 minute is used.

5.8 quit=

The quit= command specifies Gfire termination conditions. The following options are recognized:

```
quit=<n> acres
quit=<n> cells
quit=<n> days
quit=<n> hours
quit=<n> minutes
quit=<n> seconds
```

where <n> is a nonzero positive integer.

Gfire constantly tests the simulation status and determines if the quit condition has been met. If not, a percent completion message based upon the quit condition is printed. For example, the command:

```
quit=24 hours
```

causes Gfire to print status messages during the simulation indicating the percent of 24 hours simulated at that point. When 24 hours of fire growth have been simulated, Gfire writes the fire growth output map and terminates.

If no quit= command is present in the command file, Gfire terminates the simulation when the next cell escape time is infinity. This occurs when (1) all reachable cells containing fuel have ignited or (2) when fire spread rate for all fuel types is zero.

6. Gfire Processing

6.1 Gfire Memory Usage

The Gfire binary executable file occupies approximately 250K of disk storage. At start up, memory is allocated for the following:

o 225 bytes for each fuel model referenced by the fuel map, plus 65 bytes for each fuel model particle,

o 112 bytes for each weather observation record, and

o 40 bytes for each cell in the current window.

The last item accounts for most memory usage by Gfire and is dependent upon the current window size. About 1 megabyte of memory is used by a 162 by 162 cell window. Try to use windows just large enough to meet the simulation's objectives.

6.2 Gfire Processing Output

Gfire displays progress messages as it performs the simulation. These messages report the simulation's percent completion based upon the quit= command criterion.

6.3 Gfire Output Products

Gfire 1.0 produces a single output product; a GRASS 3.0 fire growth map sized to the current working window dimensions and resolution. The output map is a standard GRASS 3.0 cell map and may be manipulated just like any other cell map.

Cells of the fire growth map actually contain the time class during which the cell first ignited. The time class size is user controlled via the period= command. Cells with a 0 class never ignited during the simulation.

Future versions of Gfire could create additional fire behavior and effects map products such as fire intensity, scorch height, and tree mortality.

7. Gfire Bugs

The following bugs and side effects are known to exist in Gfire 1.0:

- o The SUN 4 UNIX scanf() family of functions erroneously interpret the blank delimited string "00" as two numbers instead of one. Be sure to use only single zero's "0" for zero values in the fuel model parameter and the fire weather observation files.

- o Gfire 1.0 terminates when the next cell escape time is infinity. This can occur whenever fuel moisture contents exceed the extinction level for all fuels in the map.

  While not technically a bug, it is undesirable for the simulation to stop when future weather conditions could cause the fire to begin spreading again. This situation commonly occurs in the intermountain west where diurnal variation in humidity causes the fire to die down in the evening and flare up again in the afternoon.

8. Fuel Model Parameter File

8.1 File Format

Fuel model parameter are specified in an ASCII text file. There is one block of records for each fuel model. Each record block contains 6 records plus 1 record for every fuel particle type. The file may contain up to 255 standard or custom fuel models, and each fuel model may contain any number of dead or live fuel particles.

Each record contains two sections; a record label terminated with a colon (:), and a record value. The records are in free field format.

The example below shows a record block for NFFL fuel model 2:

```
MODEL: 02
LABEL: NFFL02
TITLE: Timber (Grass and Understory)
DEPTH: 1.00
DMEXT: 0.15
FUELS: 4
D 3000. 0.0920 8000. 32.0 0.0555 0.0100
D  109. 0.0460 8000. 32.0 0.0555 0.0100
D   30. 0.0230 8000. 32.0 0.0555 0.0100
L 1500. 0.0230 8000. 32.0 0.0555 0.0100
```

The MODEL: Record

The MODEL: record associates the fuel model with GRASS 3.0 fuel map cell values. In the example above, fuel map cells with a value of 2 will use NFFL02 in its fire behavior computations.

It is important that the fuel model parameter file contains a MODEL corresponding to every cell value in the fire map. Indeed, a MODEL: 0 must be present if the fuel map has 0 value cells (see MODEL: 00 in the sample file below).

MODEL and fuel cell values must be in the range [0-255], need not be consecutive, and need not be zero padded.

The LABEL: Record

The LABEL: record simply attaches a short (1-8 character) label to the model.

The TITLE: Record

The TITLE: record attaches a more complete description (1-32 characters) to the model.

The DEPTH: Record

The DEPTH: record assigns the fuel model depth in feet. This depth must be a positive nonzero number!

The DMEXT: Record

The DMEXT: record assigns the dead fuel moisture content of extinction. This value must be expressed as a fraction of the oven dry weight.

The FUELS: Record

The FUELS: record declares the number of fuel particle parameter records to follow.

Fuel Particle Parameter Records

Fuel particle parameter records contain the following values:

1 fuel condition, "L" for live, or "D" for dead,
2 surface area to volume ratio (square feet per cubic foot),
3 fuel load (pounds per cubic feet),
4 low heat of combustion (BTUs per pound),
5 particle density (pounds per cubic foot),
6 total mineral content (fraction oven dry weight), and
7 silica free mineral content (fraction oven dry weight).

The record is in free field format; values must be separated by one or more blanks or tabs.

8.2 Sample Fuel Model Parameter File

The following sample fuel model parameter file contains definitions for the 13 NFFL fire behavior fuel models plus a "no fuel" model. Custom fuel models may be added as needed.

```
MODEL: 00
LABEL: NOFUEL
TITLE: No Flammable Fuel Present
DEPTH: 0.01
DMEXT: 0.01
FUELS: 0
MODEL: 01
LABEL: NFFL01
TITLE: Short Grass (1 ft)
DEPTH: 1.00
DMEXT: 0.12
FUELS: 1
D 3500. 0.0340 8000. 32.0 0.0555 0.0100
MODEL: 02
LABEL: NFFL02
TITLE: Timber (Grass and Understory)
DEPTH: 1.00
DMEXT: 0.15
FUELS: 4
D 3000. 0.0920 8000. 32.0 0.0555 0.0100
D  109. 0.0460 8000. 32.0 0.0555 0.0100
D   30. 0.0230 8000. 32.0 0.0555 0.0100
L 1500. 0.0230 8000. 32.0 0.0555 0.0100
MODEL: 03
LABEL: NFFL03
TITLE: Tall Grass (2.5 ft)
DEPTH: 2.50
DMEXT: 0.25
FUELS: 1
D 1500. 0.1380 8000. 32.0 0.0555 0.0100
MODEL: 04
LABEL: NFFL04
TITLE: Chaparral (6 ft)
DEPTH: 6.00
DMEXT: 0.20
FUELS: 4
D 2000. 0.2300 8000. 32.0 0.0555 0.0100
D  109. 0.1840 8000. 32.0 0.0555 0.0100
D   30. 0.0920 8000. 32.0 0.0555 0.0100
L 1500. 0.2300 8000. 32.0 0.0555 0.0100
MODEL: 05
LABEL: NFFL05
TITLE: Brush (2 ft)
DEPTH: 2.00
DMEXT: 0.20
FUELS: 3
D 2000. 0.0460 8000. 32.0 0.0555 0.0100
D  109. 0.0230 8000. 32.0 0.0555 0.0100
L 1500. 0.0920 8000. 32.0 0.0555 0.0100
MODEL: 06
LABEL: NFFL06
TITLE: Dormant Brush, Hardwood Slash
DEPTH: 2.50
DMEXT: 0.25
FUELS: 3
D 1750. 0.0690 8000. 32.0 0.0555 0.0100
D  109. 0.1150 8000. 32.0 0.0555 0.0100
D   30. 0.0920 8000. 32.0 0.0555 0.0100
```

```
MODEL: 07
LABEL: NFFL07
TITLE: Southern Rough
DEPTH: 2.50
DMEXT: 0.40
FUELS: 4
D 1750. 0.0520 8000. 32.0 0.0555 0.0100
D  109. 0.0860 8000. 32.0 0.0555 0.0100
D   30. 0.0690 8000. 32.0 0.0555 0.0100
L 1550. 0.0170 8000. 32.0 0.0555 0.0100
MODEL: 08
LABEL: NFFL08
TITLE: Closed Timber Litter
DEPTH: 0.20
DMEXT: 0.30
FUELS: 3
D 2000. 0.0690 8000. 32.0 0.0555 0.0100
D  109. 0.0460 8000. 32.0 0.0555 0.0100
D   30. 0.1150 8000. 32.0 0.0555 0.0100
MODEL: 09
LABEL: NFFL09
TITLE: Hardwood Litter
DEPTH: 0.20
DMEXT: 0.25
FUELS: 3
D 2500. 0.1340 8000. 32.0 0.0555 0.0100
D  109. 0.0190 8000. 32.0 0.0555 0.0100
D   30. 0.0070 8000. 32.0 0.0555 0.0100
MODEL: 10
LABEL: NFFL10
TITLE: Timber (Litter and Understory)
DEPTH: 1.00
DMEXT: 0.25
FUELS: 4
D 2000. 0.1380 8000. 32.0 0.0555 0.0100
D  109. 0.0920 8000. 32.0 0.0555 0.0100
D   30. 0.2300 8000. 32.0 0.0555 0.0100
L 1500. 0.0920 8000. 32.0 0.0555 0.0100
MODEL: 11
LABEL: NFFL11
TITLE: Light Logging Slash
DEPTH: 1.00
DMEXT: 0.15
FUELS: 3
D 1500. 0.0690 8000. 32.0 0.0555 0.0100
D  109. 0.2070 8000. 32.0 0.0555 0.0100
D   30. 0.2530 8000. 32.0 0.0555 0.0100
MODEL: 12
LABEL: NFFL12
TITLE: Medium Logging Slash
DEPTH: 2.30
DMEXT: 0.20
FUELS: 3
D 1500. 0.1840 8000. 32.0 0.0555 0.0100
D  109. 0.6440 8000. 32.0 0.0555 0.0100
D   30. 0.7590 8000. 32.0 0.0555 0.0100
MODEL: 13
LABEL: NFFL13
TITLE: Heavy Logging Slash
DEPTH: 3.00
DMEXT: 0.25
FUELS: 3
D 1500. 0.3220 8000. 32.0 0.0555 0.0100
D  109. 1.0580 8000. 32.0 0.0555 0.0100
D   30. 1.2880 8000. 32.0 0.0555 0.0100
```

9. Weather Observation File

9.1 File Format

Weather observations are stored in an ASCII text file containing one record per weather observation. The free field format records contain the following nine values:

1 observation day (integer),
2 observation hour (integer),
3 wind speed at midflame height (miles per hour),
4 wind source (compass degrees),
5 1 hour dead fuel moisture content (percent),
6 10 hour dead fuel moisture content (percent),
7 100 hour dead fuel moisture content (percent),
8 herbaceous live fuel moisture content (percent), and
9 woody live fuel moisture content (percent).

9.2 Sample File

The following sample weather observation file contains hourly data from 13:00 hours on day 1 through 12:00 hours on day 2:

```
01 13 04 180 08 10 12 100 100
01 14 05 180 07 09 12 100 100
01 15 06 180 06 09 12 100 100
01 16 07 180 05 08 12 100 100
01 17 08 180 04 08 12 100 100
01 18 08 180 04 08 12 100 100
01 19 07 180 05 08 12 100 100
01 20 06 180 06 09 12 100 100
01 21 05 180 08 09 12 100 100
01 22 04 180 10 10 12 100 100
01 23 03 180 12 10 12 100 100
01 24 02 180 14 11 12 100 100
02 01 02 180 16 11 12 100 100
02 02 02 180 18 12 12 100 100
02 03 02 180 20 12 12 100 100
02 04 02 180 22 13 12 100 100
02 05 02 180 24 13 12 100 100
02 06 02 180 26 14 12 100 100
02 07 02 180 26 14 12 100 100
02 08 02 180 22 13 12 100 100
02 09 02 180 18 12 12 100 100
02 10 02 180 16 12 12 100 100
02 11 03 180 12 11 12 100 100
02 12 04 180 08 11 12 100 100
```

10. Sample Fire Growth Output Maps

10.1 How They Were Made

Two sample fire growth output maps are presented. Fuels and terrain layers for the southeast corner of Yellowstone National Park were used as inputs. The window was 18 kilometers west to east and 12 kilometers south to north. Cell resolution was 50 meters, resulting in 86,400 cells. Gfire computation time for each map was about 50 minutes on an unshared SUN 4 system.

The first map used a single weather observation to drive fire growth with constant weather over a 24 hour period.

The second map used the sample weather file listed in Section 9.2 to drive fire growth under diurnal weather conditions over a 24 hour period.

Both fire growth output maps were then "patch"ed onto the aspect layer to illustrate fire growth through local terrain. The underlying aspect shows through unburned areas and are drawn in grey scale. Colored areas illustrate 1 hour growth classes.

11. Acknowledgements

This research was supported in part by funds provided by the Intermountain Research Station, Forest Service, U.S. Department of Agriculture (Agreement No. INT-89417-RJVA), and by the Bureau of Land Management, U.S. Department of Interior (Agreement No. AA851-IAO-77).

Additional funding and support was provided by Systems for Environmental Management, a Missoula, Montana nonprofit research and educational organization.

Gfire Version 1.0

A Fire Growth Simulator for GRASS 3.0

Technical Documentation

November 1989

Collin D. Bevins
Systems for Environmental Management
P.O. Box 8868
Missoula, MT 59806

0. Table of Contents

1.0 Introduction

Gfire 1.0 is a wildland fire growth simulator integrated into the GRASS 3.0 geographic information system.

GRASS 3.0 is a general purpose GIS developed by the US Army Corps of Engineers CERL. GRASS is written in the C programming language and runs under the UNIX operating system. It is a public domain open architecture system with published interface libraries. This makes GRASS 3.0 a fertile environment for developing specialized GIS based applications such as fire behavior and growth simulators.

Gfire is a first attempt to interface established fire behavior algorithms with a widely used geographic information system. It too is written in the C programming language and makes extensive use of the GRASS 3.0 interface libraries. The major objective of the development project was to demonstrate the feasibility of integrating fire behavior models with a GIS.

Gfire modules were first coded, compiled, and debugged using the Borland Turbo C 2.00 (tm). The integrated editor, compiler, and debugger provided a greatly enhanced development environment. Source code was then transferred to the SUN 4 (tm) computer hosting the GRASS 3.0 software where it was recompiled, linked, and tested. Gfire source code has therefore passed compiler and linker testing in two different environments with no changes in source code.

2.0 C Standards and Portability Considerations

Gfire modules are written in ANSI standard C. Only those ANSI standard library functions available on UNIX systems are referenced. Global names up to 32 characters in length are used to enhance readability. The 'old' style of declaring function parameters is used for compatibility with the SUN 4 UNIX C compiler. The Turbo C version, however, does include ANSI style function prototyping to ensure the correct number and type of function parameters and return values.

Gfire source code is identical for both the Turbo C and SUN 4 versions. To compile the Turbo C source code on the SUN 4, the definition of "#define TC" on line 11 of gfire.h must be removed.

3.0 Compilation Considerations

Under the GRASS 3.0 development guidelines, each tool has its own source directory containing C source, header files, a prototype make file ("Gmakefile"), and a complete make file ("makefile"). The Gfire source code is contained in directory /usr/grass3/src/Gfire/.

The GRASS 3.0 script /usr/grass3/src/CMD/Gmake compiles and links Gfire modules. Gmake reads the prototype Gmakefile file, creates "makefile", and compiles and/or links Gfire modules as necessary. The resulting binary file is /usr/grass3/bin/Gfire.

4.0 Memory Considerations

Gfire 1.0 uses the current GRASS 3.0 window for fire growth simulation. The fuel map, ignition map, and elevation map are all clipped to the window. Gfire dynamically allocates an internal F_CELL map of size window.rows by window.cols. Each F_CELL struct in the two-dimensional array requires 40 bytes. An error message is generated if the window is too large for available memory.

5.0 Source Modules

Gfire Version 1.0 consists of 10 C modules:

o	elev.c	Processes the GRASS 3.0 elevation input map
o	fuels.c	Processes the fuel model input file
o	Gfire.c	Main driver program
o	growth.c	Simulates fire growth over time
o	ignition.c	Processes the GRASS 3.0 fire ignition input map
o	inputs.c	Processes the Gfire command file
o	map.c	Allocates internal fire map memory space
o	outputs.c	Creates GRASS 3.0 fire growth output maps
o	rothermel.c	Computes fuel model combustion and current fire behavior values
o	weather.c	Processes the weather input file

The Turbo C development version of Gfire 1.0 requires surrogate a GRASS 3.0 module to avoid link errors:

o	grass3.c	GRASS 3.0 surrogate functions for Turbo C

All Gfire modules include a Gfire and a GRASS 3.0 header file:

o	gis.h	GRASS 3.0 header file (which references "gisdefs.h")
o	gfire.h	Gfire 1.0 macros, typedefs, globals, and function prototypes

6.0 Function Prototypes

elev.c

```
void Gfire_load_elevation_map(void);
int Gfire_neighbor(int row, int col, int dir, int *nrow, int *ncol);
```

fuels.c

```
F_MODEL *Gfire_alloc_model(int particles);
void Gfire_free_model(F_MODEL *f_model_ptr);
void Gfire_list_all_models(FILE *file_ptr);
void Gfire_list_model(F_MODEL *model_ptr, FILE *file_ptr);
void Gfire_load_fuel_map(void);
void Gfire_load_models_file(void);
int  Gfire_read_model_file(uchar *file_name);
static void Gfire_read_line(FILE *file_ptr, char *name, char *value);
static void Gfire_combust_models(void);
```

Gfire.c

```
void main(int argc, char **argv);
```

growth.c

```
void  Gfire_simulate_fire(void);
static double slope(F_CELL *cell, F_CELL *neighbor, unsigned vec);
```

inputs.c

```
void  Gfire_get_inputs(int argc, char **argv);
char *Gfire_get_fuel_mapname(void);
char *Gfire_get_ign_mapname(void);
char *Gfire_get_elev_mapname(void);
char *Gfire_get_spread_mapname(void);
char *Gfire_get_intensity_mapname(void);
char *Gfire_get_scorch_mapname(void);
char *Gfire_get_weather_filename(void);
char *Gfire_get_model_filename(void);
static void Gfire_read_name_value(char *file_name, NVP *name_value);
static void Gfire_scan_command_line(int argc, char **argv);
static int  Gfire_stash_parms(int position, char *value);
static void Gfire_verify_command_presence(void);
```

ignition.c

```
void Gfire_load_ignition_map(void);
```

map.c

```
void Gfire_allocate_map(void);
char **dim2(int rows, int cols, unsigned size);
void free2(char **prow);
F_CELL *Gfire_get_neighbor(unsigned row, unsigned col, unsigned vec) ;
void Gfire_load_window(void);
```

outputs.c

```
void Gfire_output_maps(void);
void Gfire_output_spread_map(char *mapname);
```

rothermel.c

```
void Gfire_behavior(F_MODEL *model_ptr, double *class_moisture, double slope, double
wind);
void Gfire_behavior_wind_slope(F_MODEL *model_ptr, double wind, double slope);
void Gfire_calc_all_behavior(WEATHER *w_ptr);
void Gfire_combustion(F_MODEL *model_ptr);
void Gfire_list_behavior(F_MODEL *model_ptr, FILE *file_ptr);
void Gfire_list_all_behavior(FILE *file_ptr);
```

weather.c

```
WEATHER *Gfire_get_weather(TIME t);
void Gfire_load_weather_file(void);
static void Gfire_calc_wind_vectors(double speed, double dir, double *vec);
static void Gfire_read_weather(FILE *fptr);
static void Gfire_weather_error(int line, char *msg);
```

7.0 Function Heirarchy

```
main
 │
 ├──printf
 │
 ├──G_gisinit
 │   │
 │   ├──strcpy,access,fopen,fgets,sscanf,strcmp
 │   ├──fclose,ceil,sizeof
 │   │
 │   └──G_fatal_error
 │       │
 │       └──printf,exit
 │
 ├──Gfire_get_inputs
 │   │
 │   ├──Gfire_scan_command_line
 │   │   │
 │   │   ├──exit
 │   │   │
 │   │   ├──G_parse_command
 │   │   │
 │   │   └──G_parse_command_usage
 │   │       │
 │   │       └──printf
 │   │
 │   ├──G_fatal_error p1
 │   │
 │   ├──Gfire_read_name_value
 │   │   │
 │   │   ├──access,fopen,fgets,strtok,strcmp
 │   │   ├──strncpy,fclose
 │   │   │
 │   │   └──G_fatal_error p1
 │   │
 │   └──Gfire_verify_command_pres
 │       │
 │       ├──sprintf,sscanf
 │       │
 │       ├──G_warning
 │       │   │
 │       │   └──printf
 │       │
 │       └──G_fatal_error p1
 │
 ├──Gfire_load_window
 │   │
 │   ├──sqrt
 │   │
 │   └──G_get_window
 │       │
 │       └──memcpy,sizeof
 │
 ├──Gfire_allocate_map
 │   │
 │   ├──sizeof
 │   │
 │   ├──dim2
 │   │   │
 │   │   └──calloc,malloc,sizeof
 │   │
 │   ├──G_fatal_error p1
 │   │
 │   └──G_allocate_cell_buf
 │       │
 │       ├──calloc,sizeof
 │       │
 │       └──G_fatal_error p1
 │
```

```
│
├──Gfire_load_fuel_map
│   │
│   ├──sprintf,exit,memset,sizeof
│   │
│   ├──Gfire_get_fuel_mapname
│   │
│   ├──G_find_cell
│   │   │
│   │   └──strcpy,strcat,access
│   │
│   ├──G_fatal_error p1
│   │
│   ├──G_open_cell_old
│   │   │
│   │   └──strcpy,strcat,open
│   │
│   ├──G_zero_cell_buf
│   │   │
│   │   └──memset,sizeof
│   │
│   ├──G_get_map_row
│   │   │
│   │   ├──lseek,sizeof,read
│   │   │
│   │   └──G_fatal_error p1
│   │
│   └──G_close_cell
│       │
│       └──close
│
├──Gfire_load_models_file
│   │
│   ├──Gfire_get_model_filename
│   │
│   ├──Gfire_read_model_file
│   │   │
│   │   ├──access,fopen,fgets,sscanf,strncmp
│   │   ├──strncpy,fclose
│   │   │
│   │   ├──G_fatal_error p1
│   │   │
│   │   ├──Gfire_read_line
│   │   │   │
│   │   │   ├──fgets,strncmp,strlen,sprintf,strcpy
│   │   │   │
│   │   │   └──G_fatal_error p1
│   │   │
│   │   ├──Gfire_free_model
│   │   │   │
│   │   │   └──free
│   │   │
│   │   └──Gfire_alloc_model
│   │       │
│   │       ├──calloc,sizeof
│   │       │
│   │       └──G_fatal_error p1
│   │
│   └──Gfire_combust_models
│       │
│       ├──printf
│       │
│       ├──Gfire_combustion
│       │   │
│       │   └──memset,sizeof,exp,min,pow
│       │
│       └──G_fatal_error p1
│
```

```
|
├─Gfire_load_elevation_map
│  |
│  ├─sprintf,exit
│  ├─Gfire_get_elev_mapname
│  ├─G_find_cell p2
│  ├─G_fatal_error p1
│  ├─G_open_cell_old p2
│  ├─G_zero_cell_buf p2
│  ├─G_get_map_row p2
│  └─G_close_cell p2
├─Gfire_load_ignition_map
│  |
│  ├─sprintf,exit,min
│  ├─Gfire_get_ign_mapname
│  ├─G_find_cell p2
│  ├─G_fatal_error p1
│  ├─G_open_cell_old p2
│  ├─G_zero_cell_buf p2
│  ├─G_get_map_row p2
│  └─G_close_cell p2
├─Gfire_load_weather_file
│  |
│  ├─access,fopen,fclose
│  ├─Gfire_get_weather_filenam
│  ├─G_fatal_error p1
│  └─Gfire_read_weather
│     |
│     ├─fgets,sscanf,calloc,sizeof
│     ├─Gfire_weather_error
│     │  |
│     │  ├─sprintf
│     │  └─G_fatal_error p1
│     ├─G_fatal_error p1
│     └─Gfire_calc_wind_vectors
│        |
│        └─max,cos
|
│
├─Gfire_simulate_fire
│  │
│  ├─time,min,max,printf
│  │
│  ├─Gfire_calc_all_behavior
│  │  │
│  │  └─Gfire_behavior
│  │     │
│  │     ├─exp,max
│  │     │
│  │     └─Gfire_behavior_wind_slope
│  │        │
│  │        └─pow,sqrt
│  │
│  ├─Gfire_get_neighbor
│  │
│  ├─Gfire_behavior_wind_slope p4
│  │
│  └─slope
│     │
│     └─max
│
└─Gfire_output_maps
   │
   ├─Gfire_get_spread_mapname
   │
   └─Gfire_output_spread_map
      │
      ├─memset,sizeof,strcpy,sprintf
      │
      ├─G_legal_filename
      │
      ├─G_fatal_error p1
      │
      ├─G_open_cell_new
      │  │
      │  └─strcpy,strcat,fopen,fileno
      │
      ├─G_zero_cell_buf p2
      │
      ├─G_put_map_row
      │  │
      │  └─write
      │
      ├─G_close_cell p2
      │
      ├─G_read_cats
      │  │
      │  └─calloc,sizeof
      │
      ├─G_mapset
      │
      ├─G_set_cats_title
      │
      ├─G_set_cat
      │
      ├─G_write_cats
      │
      └─G_free_cats
         │
         └─free
```

```
Gfire_stash_parms
 └──strcmp,strlen,strcpy

int
 └──strncmpi,strlen

Gfire_list_all_behavior
 └──fprintf

Gfire_list_all_models
 └─Gfire_list_model
     └──fprintf

Gfire_list_behavior
 └──fprintf

free2
 └──free

G_window_cols

G_window_rows

Gfire_get_intensity_mapna

Gfire_get_scorch_mapname

Gfire_get_weather
```

INDEX of Tree Diagram pages where routine appears.

Routine	Definition File	Definition Line	Tree Diagram Page				
access			1	2	3	4	
calloc			2	3	4	5	
ceil			1				
close			2				
cos			4				
dim2	MAP.C	53	2				
exit			1	2	3		
exp			3	4			
fclose			1	3	4		
fgets			1	3	4		
fileno			5				
fopen			1	3	4	5	
fprintf			5				
free			3	5	6		
free2	MAP.C	83	6				
Gfire_alloc_model	FUELS.C	35	3				
Gfire_allocate_map	MAP.C	29	2				
Gfire_behavior	GFIRE.H	232	4				
Gfire_behavior_wind_	ROTHERML.C	182	4				
Gfire_calc_all_behav	ROTHERML.C	237	4				
Gfire_calc_wind_vect	WEATHER.C	55	4				
Gfire_combust_models	FUELS.C	92	3				
Gfire_combustion	ROTHERML.C	259	3				
Gfire_free_model	FUELS.C	134	3				
Gfire_get_elev_mapna	INPUTS.C	146	3				
Gfire_get_fuel_mapna	INPUTS.C	140	2				
Gfire_get_ign_mapnam	INPUTS.C	143	3				
Gfire_get_inputs	INPUTS.C	110	1				
Gfire_get_intensity_	INPUTS.C	152	6				
Gfire_get_model_file	INPUTS.C	161	2				
Gfire_get_neighbor	MAP.C	97	4				
Gfire_get_scorch_map	INPUTS.C	155	6				
Gfire_get_spread_map	INPUTS.C	149	4				
Gfire_get_weather	WEATHER.C	94	6				
Gfire_get_weather_fi	INPUTS.C	158	4				
Gfire_list_all_behav	ROTHERML.C	443	5				
Gfire_list_all_model	FUELS.C	178	5				
Gfire_list_behavior	ROTHERML.C	479	5				
Gfire_list_model	FUELS.C	193	5				
Gfire_load_elevation	ELEV.C	25	3				
Gfire_load_fuel_map	FUELS.C	246	2				
Gfire_load_ignition_	IGNITION.C	19	3				
Gfire_load_models_fi	FUELS.C	303	2				
Gfire_load_weather_f	WEATHER.C	113	4				
Gfire_load_window	MAP.C	116	1				
Gfire_output_maps	OUTPUTS.C	26	4				
Gfire_output_spread_	OUTPUTS.C	53	5				
Gfire_read_line	FUELS.C	481	3				
Gfire_read_model_fil	FUELS.C	336	2				
Gfire_read_name_valu	INPUTS.C	169	1				
Gfire_read_weather	WEATHER.C	153	4				
Gfire_scan_command_l	INPUTS.C	236	1				
Gfire_simulate_fire	GROWTH.C	31	4				
Gfire_stash_parms	INPUTS.C	263	5				
Gfire_verify_command	INPUTS.C	304	1				
Gfire_weather_error	WEATHER.C	242	4				
G_allocate_cell_buf	GRASS3.C	60	2				
G_close_cell	GRASS3.C	72	2	3	4	5	
G_fatal_error	GRASS3.C	78	1	2	3	4	5
G_find_cell	GRASS3.C	87	2	3			
G_free_cats	GRASS3.C	99	5				
G_get_map_row	GRASS3.C	108	2	3	4		
G_get_window	GRASS3.C	126	2				
G_gisinit	GRASS3.C	135	1				
G_legal_filename	GRASS3.C	189	5				

G_mapset	GRASS3.C	195	5				
G_open_cell_new	GRASS3.C	201	5				
G_open_cell_old	GRASS3.C	214	2	3	4		
G_parse_command	GRASS3.C	226	1				
G_parse_command_usag	GRASS3.C	244	1				
G_put_map_row	GRASS3.C	250	5				
G_read_cats	GRASS3.C	258	5				
G_set_cat	GRASS3.C	268	5				
G_set_cats_title	GRASS3.C	274	5				
G_warning	GRASS3.C	280	1				
G_window_cols	GRASS3.C	292	6				
G_window_rows	GRASS3.C	286	6				
G_write_cats	GRASS3.C	298	5				
G_zero_cell_buf	GRASS3.C	304	2	3	4	5	
int	GRASS3.C	227	5				
lseek			2				
main	GFIRE.C	13	1				
malloc			2				
max			4				
memcpy			2				
memset			2	3	5		
min			3	4			
open			2				
pow			3	4			
printf			1	3	4		
read			2				
sizeof			1	2	3	4	5
slope	GROWTH.C	251	4				
sprintf			1	2	3	4	5
sqrt			2	4			
sscanf			1	3	4		
strcat			2	5			
strcmp			1	5			
strcpy			1	2	3	5	
strlen			3	5			
strncmp			3				
strncmpi			5				
strncpy			1	3			
strtok			1				
time			4				
write			5				

8.0 Gmakefile

```
# %W% %G%
PGM = Gfire

LIST =  Gfire.o\
        elev.o\
        fuels.o\
        growth.o\
        ignition.o\
        inputs.o\
        map.o\
        outputs.o\
        rothermel.o\
        weather.o

$(BIN)/$(PGM): $(LIST) $(GISLIB)
        cc $(LDFLAGS) $(LIST) $(GISLIB) $(MATHLIB) -o $@

$(LIST): gfire.h

$(GISLIB):      #in case library changes
```

9.0 makefile

```
GMAKE=/usr/grass3/src/CMD/Gmake
# set these definitions according to your system requirements

GIS                  = /usr/grass3
GISDBASE             = /usr/grass3/data
UNIX_BIN             = /usr/local/bin
DEFAULT_LOCATION     = spearfish

#OS                   = SYSV
OS                   = BERK
COMPILE_FLAGS        = -g
LDFLAGS              =
DIGIT_FLAGS          =
#DIGIT_FLAGS          = -DATT
#DIGIT_FLAGS          = -DMASSCOMP
MATHLIB              = -lm
TERMLIB              = -ltermlib
CLEAR                = ok
#CLEAR                = no
AR                   = ar ruv $@ $?; ranlib $@
#AR                   = ar rc $@ `lorder $(OBJ) | tsort`
####################################################CFLAGS          =  $(COMPILE_FLAGS)  -I$(LIBDIR)  -D$(OS)
$(EXTRA_CFLAGS)
GMAKE       = $(GIS)/src/CMD/Gmake
MAKEALL     = set - *; for d do test -f $$d/Gmakefile && $(GMAKE) $$d; done; exit 0
MANROFF     = tbl -TX $(GIS)/src/man.help/man.header $? | nroff -Tlp | col -b > $@
MAN1        = $(GIS)/man/1
MAN2        = $(GIS)/man/2
HELP        = $(GIS)/man/help

BIN         = $(GIS)/bin
ETC         = $(GIS)/etc
SRC         = $(GIS)/src
LIBDIR      = $(GIS)/src/libes
GISLIB      = $(LIBDIR)/libgis.a
VASKLIB     = $(LIBDIR)/libvask.a
LOCKLIB     = $(LIBDIR)/liblock.a
IMAGERYLIB  = $(LIBDIR)/libI.a
SEGMENTLIB  = $(LIBDIR)/libsegment.a
DLGLIB      = $(LIBDIR)/libdlg.a
CURSES      = -lcurses $(TERMLIB)
VASK        = $(VASKLIB) $(CURSES)
RASTERLIB   = $(SRC)/D/libes/rasterlib.a
DISPLAYLIB  = $(SRC)/D/libes/displaylib.a
D_LIB       = $(SRC)/D/libes/Dlib.a
#################################################### %W% %G%
PGM = Gfire

LIST =  Gfire.o\
        elev.o\
        fuels.o\
        growth.o\
        ignition.o\
        inputs.o\
        map.o\
        outputs.o\
        rothermel.o\
        weather.o

$(BIN)/$(PGM): $(LIST) $(GISLIB)
        cc $(LDFLAGS) $(LIST) $(GISLIB) $(MATHLIB) -o $@

$(LIST): gfire.h

$(GISLIB):      #in case library changes
```

```
#####################################################
# the tags file created by "make ctags" is great for
# editting to look for function definitions
# vi -t <function name>

ctags:
	ctags *.[ch]
	sed 's/\?/\//g' tags > tags.tmp
	mv tags.tmp tags

# this next rule builds .s files from .o files preserving all the CFLAGS
# to make obj.s simply enter "make obj.s"

.c.s:
	cc $(CFLAGS) -S $<
```

10. Algorithms

10.1 Fire Growth Algorithm

```
Set all cell ignition times to infinity.
Read first weather observation.
Calculate wind speed in 8 directions.
Calculate combustion variables for all fuel models.
Set next_ignition_time to first source ignition time.
Determine quit conditions.

While next_ignition_time < infinity...
    Set previous_time to current_time.
    Set current_time to next_ignition_time.
    Set next_ignition_time to infinity.

    If current_time exceeds current weather period...
        Set next_time to current_time.
        Set current_time to current weather period's last time.
        Set new_weather_flag and continue.

    For each row and column in the current window...
        If cell_ignition_time > current_time, loop to next column or row.
        If cell has no fuel, loop to next column or row.
        If all cell's neighbors are ignited, loop to next column or row.

        For each of cell's 8 neighbors...
            If neighbor_ignition_time <= current_time, loop to next neighbor.
            If neighboring cell has no fuel, loop to next neighbor.
            Set next_time = minimum(next_time, neighbor_ignition_time).
            Calculate slope from source to neighboring cell.
            Calculate spread rate with wind and slope from source cell to neighbor.
            Update distance fire has traveled in this vector since previous_time.
            Calculate time to ignite neighbor under current spread conditions.

            If new_neighbor_ignition_time < neighbor_ignition_time...
                Set neighbor_ignition_time = new_neighbor_ignition_time.
                Set next_ignition_time = minimum(next_ignition_time, neighbor_ignition_time).
        Next neighboring cell.
    Next column and row.

    Display progress message to console.
    If quit conditions have been met (time or size)...
        Print completion summary.
        Return to calling program.

    If weather_flag is set...
        Read next weather observation.
        Recalculate wind speed in 8 directions.
        Recalculate no-wind, no_slope fire behavior for all fuel models.
        Clear weather_flag.
        Set next_time to current_time to force weather condition update.
Continue with next_time.
```

10.2 Wind Speed Algorithm

Gfire 1.0 uses a simple cosine function in Gfire_calc_wind_vectors(float wind_speed) to calculate wind speed in eight directions.

First the wind source (compass degrees) is converted to wind bearing (compass degrees). The wind speed in each of the eight 45 degree compass points is then calculated as the wind speed at the wind bearing angle multiplied by the maximum of (1) zero or (2) the cosine of the difference between the wind bearing angle and the compass angle:

```
wind_speed(compass_angle) =
        wind_speed(bearing_angle) * maximum(0., (bearing_angle - compass_angle) )
```

This method may be easily replaced by other algorithms.

11. Acknowledgements

This research was support in part by funds provided by the Intermountain Research Station, Forest Service, U.S. Department of Agriculture (Agreement No. INT-89417-RJVA), and by the Bureau of Land Management, U.S. Department of Interior (Agreement No. AA851-IAO-77).

Additional funding and support was provided by Systems for Environmental Management, a Missoula, Montana nonprofit research and educational corporation.

12. Module Source Listings and Cross Reference

Appended